ANALYSING SALES DATA AT ORION'S STAR

ASHIRI THENUKA

DAMILOLA TIJANI

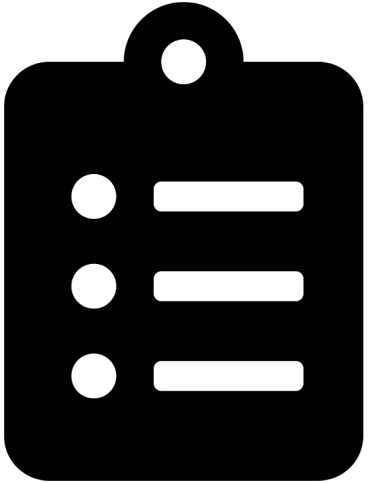
AMISHA SHETTY

MUZIB SHEIK

SANGHAVI PENDEM

TABLE OF CONTENT

1. Overview
2. Problem Statement
3. Business Questions
4. Methodology
5. Data Extraction
6. Data Transformation
7. Data Loading
8. Data Visualisation
9. Business Reports
10. Recommendation
11. Reference
12. Appendix



Overview

Orion Star is a global retailer that offers a range of sports and outdoor goods in various nations. Due to a variety of circumstances, each product's level of success will vary between nations. Therefore, it is crucial to comprehend these aspects and how they affect sales in order to support the expansion of the company. Big data analytics will be utilised to analyse each country's market and make wise decisions about marketing and business growth.

Problem Statement

“We want to be able to understand the data we gather from all branches so we can know what aspects of the business that need to be worked on, how we can maximize sales in different regions and subsequently grow the business ”

The management at Orion Star would like to gain meaningful insights from their data that can be used to improve sales and the business.

With the proposed system, Orion Star will be better able to gain actionable knowledge from the data and answer the various business question concerning product sales, supply, profits, and many other aspects of the business.

Business Questions?

S. No	Business Question	Justification	Metric	Dimensions	Lowest level of granularity
1	Who were the last month's top ten customers by sales value?	Recognizing most valuable customers will help in customer retention strategies	Total Retail price	Time, Customer	Time – Month, Customer - Individual
2	What was the most profitable product of 2002?	Most Profitable products will be measured, and production of these products will be planned accordingly	Profit	Time, Product	Time – Year, Product - Product item
3	What was the total number of products purchased in 2002 by customer type?	When a customer type is established to have the highest number of products purchased, focus can be placed on that group	Quantity	Time, Product, Customer	Time – Year, Product – Product item, Customer – Customer type
4	Which customer type has generated the least amount of income in 2002?	For the customers with least purchase graph, measures such as new offers will be created to attract the customers	Profit	Time, Customer	Time – Year, Customer – Customer Type
5	What were the total sales over time by country by product group?	Identifying most sales based on demographic location and product group will give help in item production and marketing	Total Retail Price	Time, Region, Product	Time – Year, Region – Country, Product – Product Group

Proposed Solution

The proposed solution for Orion Sword (coined from Orion Star and will be used to refer to the project subsequently) is to build a Data warehouse.

A data warehouse is a type of data management system built to facilitate and support business intelligence activities, particularly analytics (Golfarelli, Matte and Stefano Rizzi, 2009). The data from Orion Star will be extracted, cleaned, transformed, and stored in Data Marts to enable easy access to data and understand the relationship and trends across each product sales.

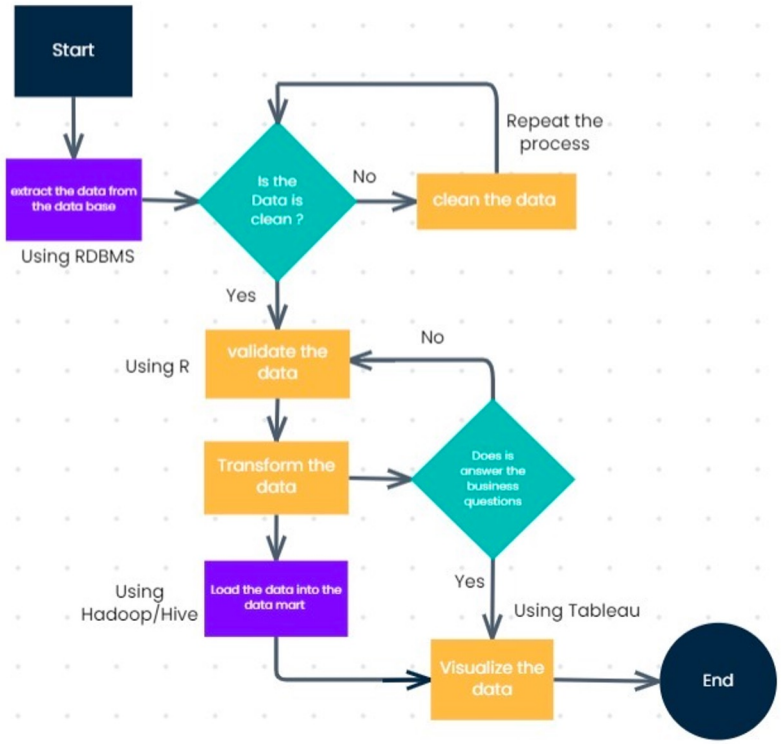


Figure 1 depicts the flow of the project development.

Proposed Development Tools

The proposed solution will help the business leverage the use of Data warehousing by building a Sales Data Mart that will answer the business questions. This solution will be built using the following business tools: Rstudio, Hive, Hadoop and Tableau.

The Data pipeline involves data going from Rstudio into Hadoop and then to Tableau for data visualisation. To optimise performance, enable steady and quick data transfers from R to Hadoop, and give Tableau users access to data in Hadoop, the data were merged using the Cloudera ODBC connection.

R programming language is used for data extraction, quality checks, cleaning, validations, and transformation because it has more small packages and better built-in functionality for data analysis than Python, which is also widely used in Big data (Hackenberger, 2020). It also allows for the writing of simpler, more manageable functions.

The star schema in Apache Hive was utilised to develop the Dimensional model since it is the greatest option for working with historical data. It offers the best results when querying a particular business criteria (Yin Huai et al., 2014). Hadoop, which is effective at storing and analysing enormous datasets, is the foundation for Apache Hive, a distributed database system. Hadoop runs using Map-reduce which is an efficient, dependable, and fault-tolerant approach for processing huge amounts of data in parallel (Yin Huai et al., 2014). This also makes it easy to scale up the product in the future.

Tableau was utilised for graphic visualisation and solution representation since it can process enormous volumes of data rapidly and without lagging (Kristi M. et al, 2012).

Development Methodology

The proposed system will be developed using the business questions. The data warehouse will be built from the bottom-up following the Kimball approach.

Ralph Kimball's approach compared to the Bill Inmon's Top-down approach builds the data marts first to facilitate the analysis of the data (Salim K. et al, 2020). These data marts are then combined to build a data warehouse. This method uses the Extract Transform Load (ETL) development method that brings the data from the different sources, transforms it by cleaning, merging etc and loads the data into a common area called the Storing area.

This method was preferred because it is more cost effective and will essentially take lesser time to be built. It also focuses on individual business area which is good for my team as we already have business questions to focus on answering for the business.

Figure 2 represents the pictorial representation of the Kimball approach. We will be going into details of each process of the development in the following slides.

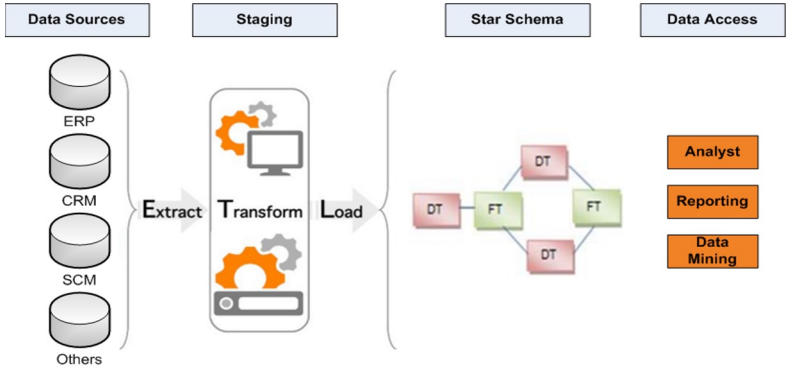


Figure 2. Kimball Approach

Data Extraction

The Data will be extracted from Orion Star's existing database. This data has been transformed into a relational database and normalised by the internal Information Systems Department of the company.

The data extracted by the Sword team will be loaded into R software for further transformation that will involve viewing the structure of the data, missing and incomplete values, possible errors and cleaning the data before loading it into the Staging area.

Some of the data extracted are shown below.

```
> county <- read.csv("county.txt")
> View(county)
> customer <- read.csv("customer.txt")
> View(customer)
> customer_type <- read.csv("customer_type.txt")
> View(customer_type)
> discount <- read.csv("discount.txt")
> View(discount)
> discount <- read.csv("discount.txt")
> View(discount)
> geo_type <- read.csv("geo_type.txt")
> View(geo_type)
> order_item <- read.csv("order_item.txt")
> View(order_item)
```

Figure 3. Data Extraction in R

geo_type × | order × | order_item × | organization × | org_level × | postal_code × | price_list ×

Filter

	Product_ID	Product_Name	Supplier_ID	Product_Level	Product_Ref_ID
1	210000000000	Children	.	4	.
2	210100000000	Children Outdoors	.	3	210000000000
3	210100100000	Outdoor things, Kids	.	2	210100000000
4	210100100001	Boy's and Girl's Ski Pants with Braces	50	1	210100100000
5	210100100002	Children's Jacket	4742	1	210100100000

Figure 4. Product

Data Extraction

geo_type × order × order_item ×

Filter

	Product_Level	Product_Level_Name
1	1	Product
2	2	Product Group
3	3	Product Category
4	4	Product Line

Figure 5. Product Level

	Country	Country_Name	Population	Country_ID	Continent_ID	Country_Former_Name
1	AQ	Antarctica	.	11	90	
2	PR	Puerto Rico	.	72	91	
3	VI	Virgin Islands (U.S.)	.	78	91	
4	AW	Aruba	.	100	91	
5	BS	Bahamas	.	180	91	
6	BM	Bermuda	.	195	91	
7	BZ	Belize	.	227	91	British Honduras
8	VG	British Virgin Islands	.	231	91	
9	CA	Canada	.	260	91	
10	KY	Cayman Islands	.	268	91	

Figure 6. Country

nStar.R* × city × continent × country × county × customer × customer_type × discount × geo_type × order × order_iter

Filter

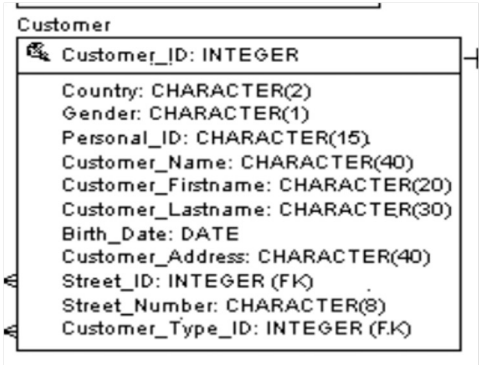
	County_ID	County_Type	County_Name	Province_Type	Province_Name	Region_Name	Region_Type	State_ID	Country
1	35000000	.		350001	N/A	N/A	350002	3500000	
2	35000001	.		350001	Ain	Rh<f4>ne-Alpes	350002	3500000	
3	35000002	.		350001	Aisne	Picardie	350002	3500000	
4	35000003	.		350001	Allier	Auvergne	350002	3500000	
5	35000004	.		350001	Alpes-de-Haute-Provence	Provence-Alpes-C<f4>te-d'Azur	350002	3500000	

Figure 7. County

Key features of the chosen data sets

Customer Dataset : Orion Star: Sports & Outdoors Database Dump.

Following dataset contains Orion Star s Customer details. This data is gathered using the purchase history of a customer.



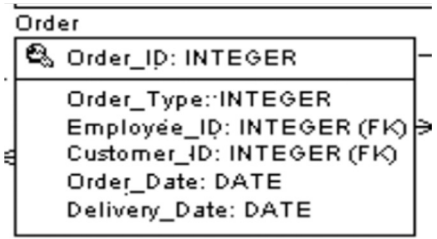
Column	Description and Data type
Customer_ID	Customer unique identification No (int)
Country	Customer Origination (String)
Gender	Customers gender (String)
Personal_ID	
Customer_Name	Name of the Customer (String)
Customer_FirstName	Customer first name (String)
Customer_LastName	Customer last name (String)
Birth_Date	Customer birthdate (Date)
Customer_Address	Customer address (String)
Street_ID	Street id of the entered address (int)
Street_Number	Street number of the entered address (int)
Customer_Type_ID	Customer Type (String)

Key features of the chosen data sets cont..

Order Dataset : Orion Star: Sports & Outdoors Database Dump.

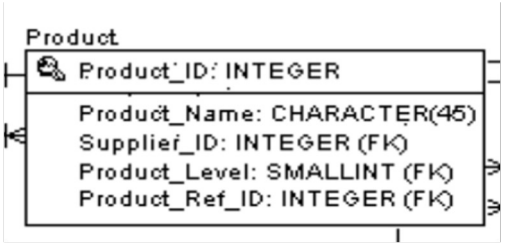
Column	Description and Data type
Order_ID	Order unique identification No (int)
Order_Type	Order Type (String)
Employee_ID	Unique employee id who processed the order (int)
Customer_ID	Unique Customer identification No (int)
Order_Date	Date which the order placed (Date)
Delivery_Date	Date which the order delivered (Date)

Table 2. Order dataset definitions



Key features of the chosen data sets cont..

Product Dataset : Orion Star: Sports & Outdoors Database Dump.



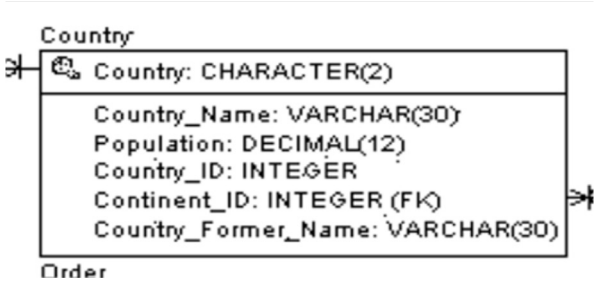
Column	Description and Data type
Product_ID	Product unique identification No (int)
Product_Name	Product Name (String)
Supplier_ID	Product supplier unique identification No (int)
Product_Level	Nature of the product (string)
Product_Ref_ID	

Table 3. Product dataset definitions

Key features of the chosen data sets cont..

Country Dataset : Orion Star: Sports & Outdoors Database Dump.

Column	Description and Data type
country	Country abbreviation (string)
Country_name	Country Name (string)
Country_ID	Unique country identification No (int)
Continet_ID	Unique Continent identification No (int)
Country_former_name	Original name of the country(string)



Justification for Final Dataset

Sword extracted relevant data from the database to build a Sales data Mart. The figures below show the structure of the datasets used.

```
> str(county)
'data.frame':   3459 obs. of  9 variables:
 $ County_ID    : int  35000000 35000001 35000002 35000003 35000004 35(
 $ County_Type  : chr  "." "." "." "." ...
 $ County_Name  : chr  "" "" "" "" ...
 $ Province_Type: chr  "350001" "350001" "350001" "350001" ...
 $ Province_Name: chr  "N/A" "Ain" "Aisne" "Allier" ...
 $ Region_Name  : chr  "N/A" "Rh\xf4ne-Alpes" "Picardie" "Auvergne" ..
 $ Region_Type  : chr  "350002" "350002" "350002" "350002" ...
 $ State_ID     : int  3500000 3500000 3500000 3500000 3500000 3500000
 $ Country      : chr  "" "" "" "" ...
> str(country)
'data.frame':   236 obs. of  6 variables:
 $ Country            : chr  "AQ" "PR" "VI" "AW" ...
 $ Country_Name       : chr  "Antarctica" "Puerto Rico" "Virgin Islands
 $ Population         : chr  "." "." "." "." ...
 $ Country_ID         : int  11 72 78 100 180 195 227 231 260 268 ...
 $ Continent_ID       : int  90 91 91 91 91 91 91 91 91 91 ...
 $ Country_Former_Name: chr  "" "" "" "" ...
```

Figure 8. Dataset Structure

```
> str(product)
'data.frame':   5579 obs. of  5 variables:
 $ Product_ID    : num  210000000000 210100000000 210100100000 210100100001 2101(
 $ Product_Name  : chr  "Children" "Children Outdoors" "Outdoor things, Kids" "Bc
 $ Supplier_ID   : chr  "." "." "." "50" ...
 $ Product_Level : int  4 3 2 1 1 1 1 1 1 1 ...
 $ Product_Ref_ID: chr  "." "210000000000" "210100000000" "210100100000" ...
> str(product_level)
'data.frame':   4 obs. of  2 variables:
 $ Product_Level     : int  1 2 3 4
 $ Product_Level_Name: chr  "Product" "Product Group" "Product Category" "Product
> str(order)
'data.frame':   755173 obs. of  6 variables:
 $ Order_ID     : int  1230000029 1230000126 1230000360 1230000396 1230000400 123
 $ Order_Type   : int  3 3 3 3 3 3 3 3 3 3 ...
 $ Employee_ID  : int  99999999 99999999 99999999 99999999 99999999 99999999 999
 $ Customer_ID  : int  21991 48163 90531 94039 78431 83582 3290 7435 9587 24730 .
 $ Order_Date   : chr  "01JAN1998" "01JAN1998" "01JAN1998" "01JAN1998" ...
 $ Delivery_Date: chr  "01JAN1998" "01JAN1998" "01JAN1998" "01JAN1998" ...
> str(order_item)
'data.frame':   962884 obs. of  6 variables:
 $ Order_ID          : int  1230000029 1230000126 1230000360 1230000396 123000046
 $ Order_Item_Num    : int  1 1 1 2 1 1 1 1 1 1 ...
 $ Product_ID        : num  240100100136 220101100034 220100100459 240100400095 2
 $ Quantity          : int  3 2 2 4 1 3 1 1 1 1 ...
```

Figure 9. Dataset Structure

Dataset

```
> str(customer)
'data.frame':   89954 obs. of  12 variables:
 $ Customer_ID       : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Country           : chr  "FR" "ES" "IT" "US" ...
 $ Gender            : chr  "M" "F" "M" "M" ...
 $ Personal_ID       : logi  NA NA NA NA NA NA ...
 $ Customer_Name     : chr  "Albert Collet" "Mercedes Mart\xednez" "Pier Egidio
 $ Customer_FirstName: chr  "Albert" "Mercedes" "Pier Egidio" "James" ...
 $ Customer_LastName : chr  "Collet" "Mart\xednez" "Boeris" "Kvarniq" ...
 $ Birth_Date        : chr  "24NOV1940" "15JAN1955" "01JUL1970" "27JUN1970" ...
 $ Customer_Address  : chr  "Square Edouard Vii 1" "Edificio 2" "Via M. Di Mont
 $ Street_ID         : chr  "3500101176" "8300101034" "4800100416" "9260106519"
 $ Street_Number     : int  1 2 3 4382 6468 2 561 5 9 425 ...
 $ Customer_Type_ID  : int  2030 1010 1040 1020 2020 1030 1040 1020 2020 1040 .
> str(customer_type)
'data.frame':   8 obs. of  4 variables:
 $ Customer_Type_ID : int  1010 1020 1030 1040 2010 2020 2030 3010
 $ Customer_Type    : chr  "Orion Club members inactive" "Orion Club members lo
mbers high activity" ...
 $ Customer_Group_ID: int  10 10 10 10 20 20 20 30
 $ Customer_Group   : chr  "Orion Club members" "Orion Club members" "Orion Clu
>
```

Figure 10. Dataset Structure

To build a Sales Data Mart that will answer the business questions and give further insights into the sales of the Orion Star, Sword has used the above data sets and viewed the structure of each using Rstudio.

The customer table will tell us the order of each customer. Metadata about the order will be viewed in Order and Order Item will give us detail about each order including quantity, cost and discount. Product Table will give us detail of each product and is connected to Product level which helps with insight on each product's level. Country will be connected to Customer to tell us the country where the order happened. A join with Customer type will give detail about the category a customer falls into.

Later in the staging area these data sets will be joined and fact and dimensional tables will be created to answerer key business questions.

Data Transformation

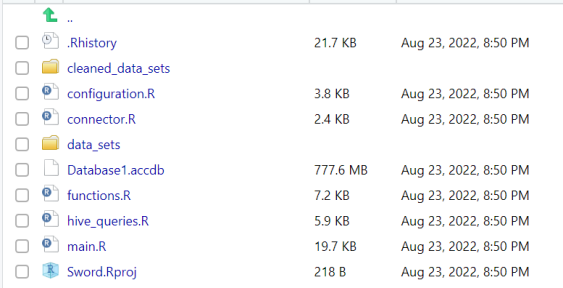
Data transformation is the process of converting the data into the suitable format we will be needing (Salim K. et al, 2020). This process involves cleaning the data by checking for incomplete/missing values, corrupted values, wrong entry data, duplicate data etc. Data is also verified and merged during the transformation process.

Data transformation takes place in the Staging area. The Staging area also involves adding data to the Operational Database system.

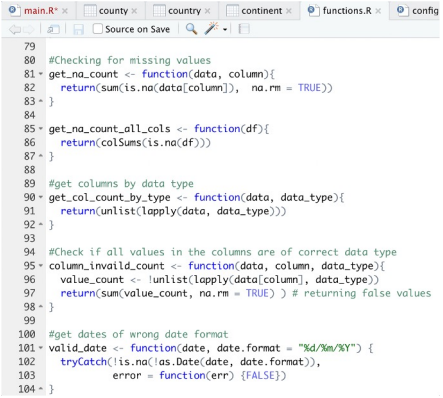
A function library was written to make scaling and editing easier. These functions contain code used for the cleaning process e.g. checking missing values, checking date formats, plotting graphs etc

The **Figures** below show the data cleaning process

Figure 12. File Structure



..		
.Rhistory	21.7 KB	Aug 23, 2022, 8:50 PM
cleaned_data_sets		
configuration.R	3.8 KB	Aug 23, 2022, 8:50 PM
connector.R	2.4 KB	Aug 23, 2022, 8:50 PM
data_sets		
Database1.accdb	777.6 MB	Aug 23, 2022, 8:50 PM
functions.R	7.2 KB	Aug 23, 2022, 8:50 PM
hive_queries.R	5.9 KB	Aug 23, 2022, 8:50 PM
main.R	19.7 KB	Aug 23, 2022, 8:50 PM
Sword.Rproj	218 B	Aug 23, 2022, 8:50 PM



```
#Checking for missing values
get_na_count <- function(data, column){
  return(sum(is.na(data[column]),  na.rm = TRUE))
}

get_na_count_all_cols <- function(df){
  return(colSums(is.na(df)))
}

#get columns by data type
get_col_count_by_type <- function(data, data_type){
  return(unlist(lapply(data, data_type)))
}

#Check if all values in the columns are of correct data type
column_invaild_count <- function(data, column, data_type){
  value_count <- !unlist(lapply(data[column], data_type))
  return(sum(value_count, na.rm = TRUE) ) # returning false values
}

#get dates of wrong date format
valid_date <- function(date, date.format = "%d/%m/%Y") {
  tryCatch(!is.na(!as.Date(date, date.format)),
           error = function(err) {FALSE})
}
```

Figure 11. Functions Library

Data Transformation continued

```r
data_set_path <- "./data_sets"
clean_data_path <-"./cleaned_data_sets"

###### city ##########
city_cols <- c( "City_ID","City_Name","Country")
###### continent ##########
continent_cols <- c( "Continent_ID","Continent_Name")
###### contry ##########
contry_cols <- c( "Country","Country_Name","Population","Country_ID","Continent_ID
contry_cols_types <- c( "character","character","integer","integer","integer","cha
######## County ######
conty_cols <- c("County_ID","County_Type","County_Name","Province_Type","Province_
conty_cols_types <- c( "integer","integer","character","integer","character","char
######## Customer ######
```

Figure 13. Configuration

```r
con <- dbConnect(odbc::odbc(),"Sample Cloudera Hive DSN",timeout=100,Schema ="sword")

dbSendQuery(con,create_city_table)
DBI::dbWriteTable(con,"city",city, overwrite=TRUE, append=FALSE)

dbSendQuery(con,create_continent_table)
DBI::dbWriteTable(con,"continent",continent, overwrite=TRUE, append=FALSE)

dbSendQuery(con,create_country_table)
DBI::dbWriteTable(con,"country",country, overwrite=TRUE, append=FALSE)

dbSendQuery(con,create_county_table)
DBI::dbWriteTable(con,"county",county, overwrite=TRUE, append=FALSE)

dbSendQuery(con,create_Customer_table)
```

Figure 14. Connector

```r
#order_item
order_item <- read.delim("order_item.txt", sep = ",", na.strings = c("", "NA"))%>%
  remove_special_charactors("Total_Retail_Price")%>%
  remove_special_charactors("Discount")
View(order_item)
sapply(order_item, class)

#Plotting missing values
plot_missing_or_invalids_or_outliers(order_item, "Customer Details", 1)
#Plotting invalid values
plot_missing_or_invalids_or_outliers(order_item, "Customer Details", 2)  #no invalid values
#checking for invalid values (negative numbers/outliers)
plot_missing_or_invalids_or_outliers(order_item, "Customer Details", 3)  #no outliers
```

Figure 15. Main

```r
#################### SQL Queries #########################
create_city_table <- 'CREATE TABLE `city`(City_ID INT,City_Name STRING, Country STRING) ROW FORMA

create_continent_table <- 'CREATE TABLE `continent`(Continent_ID INT,Continent_Name STRING) ROW F

create_country_table <- 'CREATE TABLE `country`(Country STRING,Country_Name STRING, Country_ID IN

create_county_table <- 'CREATE TABLE `county`(County_ID INT,County_Type INT, County_Name STRING,

create_Customer_table <- 'CREATE TABLE `customer`(Customer_ID INT,Country STRING,Gender STRING,Cu

create_customer_type_table <- 'CREATE TABLE `customer_type`(Customer_Type_ID INT,Customer_Type ST

create_discount_table <- 'CREATE TABLE `discount`(Product_ID INT,Start_Date DATE,End_Date DATE,Un
```

Figure 16. Hive Queries

Data Transformation continued

```r
#######################################################
######################## county ######################
#######################################################

county <- read.delim(paste(data_set_path,"county.txt",sep="/"), sep = ",", na.strings = c("", "NA")) %>%
  filter_columns(conty_cols)%>%
  set_data_t(conty_cols_types)%>%
  replace_str_vals("N/A",NA)%>%
  as_tibble()
View(county)
sapply(county, class)

#Plotting missing values
plot_missing_or_invalids_or_outliers(county, "Customer Details", 1)
#Plotting invalid values
plot_missing_or_invalids_or_outliers(county, "Customer Details", 2)  #no invalid values
#checking for invalid values (negative numbers/outliers)
plot_missing_or_invalids_or_outliers(county, "Customer Details", 3)  #no outliers

# Dropping NA columns
county <- rem_col_by_name(county,c("Province_Type","Province_Name","Region_Name","Region_Type","Country"))
county <- county %>%na.omit()
#write.csv(county,paste(clean_data_path,"county.csv",sep="/"),row.names = FALSE)
#######################################################
```

Figure 12. Data Cleaning process

main.R* × | county × | country × | continent × | functions.R × | configuration.R × | city ×

Filter

	County_ID	County_Type	County_Name	Province_Type	Province_Name	Region_Name	Region_Type	State_ID	Countr
1	35000000	NA	NA	350001	NA	NA	350002	3500000	NA
2	35000001	NA	NA	350001	Ain	Rh<f4>ne-Alpes	350002	3500000	NA
3	35000002	NA	NA	350001	Aisne	Picardie	350002	3500000	NA
4	35000003	NA	NA	350001	Allier	Auvergne	350002	3500000	NA
5	35000004	NA	NA	350001	Alpes-de-Haute-Provence	Provence-Alpes-C<f4>te-d'Azur	350002	3500000	NA
6	35000005	NA	NA	350001	Hautes-Alpes	Provence-Alpes-C<f4>te-d'Azur	350002	3500000	NA
7	35000006	NA	NA	350001	Alpes-Maritimes	Provence-Alpes-C<f4>te-d'Azur	350002	3500000	NA
8	35000007	NA	NA	350001	Ardeche	Rh<f4>ne-Alpes	350002	3500000	NA
9	35000008	NA	NA	350001	Ardennes	Champagne-Ardenne	350002	3500000	NA
10	35000009	NA	NA	350001	Ariege	Midi-Pyr<e9>n<e9>es	350002	3500000	NA

Figure 13. Dataset before cleaning

The code above extracts the data into R using read.delim and it replaces empty data with NA. A plot is drawn to view the distribution of NA values in the dataset. This is also done to view the invalids and outliers in the data set.

```
> sapply(county, class)
    County_ID   County_Type   County_Name Province_Type Province_Name   Region_Name   Region_Type      State_ID       Country
    "integer"     "integer"   "character"     "integer"   "character"   "character"     "integer"     "integer"   "character"
```

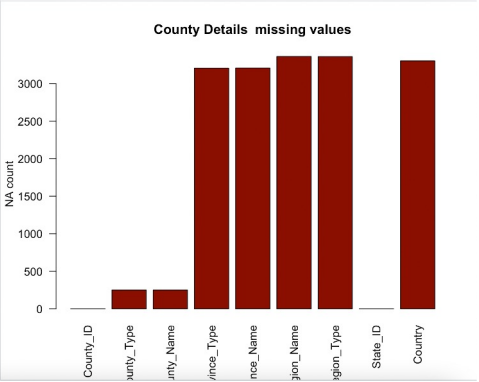
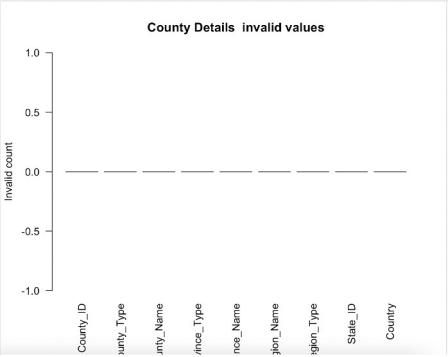


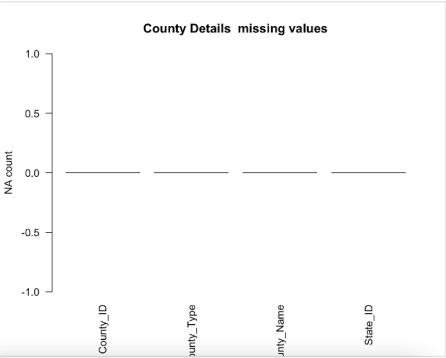
Figure 14. Distribution of NA values

Data Transformation

Variables with large amounts of missing data that do not necessarily impact or help to answer our business questions are removed using the rem_col_by_name function. This helps to name the variables to be taken out in order to have only clean and relevant data in our Data mart.

After data is cleaned, we plot graphs to validate our data. This process is carried out for every dataset used.

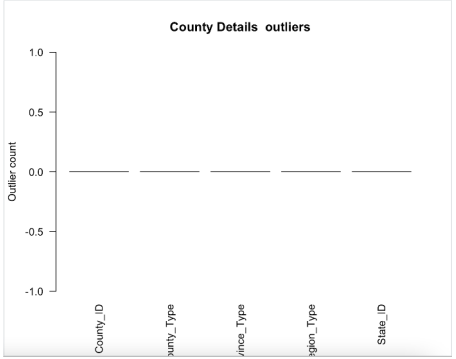




main.R* | county | country | continent | functions.R

Filter

	County_ID	County_Type	County_Name	State_ID
1	926000001	926001	Aleutians East	9260002
2	926000002	926001	Aleutians West	9260002
3	926000003	926001	Aleutians West (CA)	9260002
4	926000004	926001	Anchorage	9260002
5	926000005	926001	Bethel (CA)	9260002
6	926000006	926001	Bristol Bay	9260002
7	926000007	926001	Denali	9260002
8	926000008	926001	Dillingham (CA)	9260002
9	926000009	926001	Fairbanks North Star	9260002
10	926000010	926001	Juneau	9260002




```
> plot_missing_or_invalids_or_outliers(county, "Customer Details", 1)
  County_ID County_Type County_Name    State_ID
          0           0           0           0
```

Data Transformation

Dates were also formatted using the ‘format_date’ function as seen in the figures below.

```
discount <- read.delim(paste(data_set_path,"discount.txt",
  formate_date("Start_Date",'%d %B %Y')%>%
  formate_date("End_Date",'%d %B %Y')%>%
  remove_special_charactors("Unit_Sales_Price")%>%
  remove_special_charactors("Discount")%>%
  set_data_t(discount_cols_types) %>%
  as_tibble()
View(discount)
sapply(discount, class)
```

	Product_ID	Start_Date	End_Date	Unit_Sales_Price	Discount
1	210100100027	01MAY1999	31MAY1999	$15.99	70%
2	210100100027	01MAY2000	31MAY2000	$15.99	70%
3	210100100027	01MAY2001	31MAY2001	$16.99	70%
4	210100100027	01MAY2002	31MAY2002	$17.99	70%
5	210100100030	01AUG1999	31AUG1999	$28.99	70%

Before Date transformation

```
#Format Date
formate_date <- function(x,col,format){
  x[[col]] <- as.Date(x[[col]],format)
  return(x)
}
```

	Product_ID	Start_Date	End_Date	Unit_Sales_Price	Discount
1	210100100027	1999-05-01	1999-05-31	15.99	70
2	210100100027	2000-05-01	2000-05-31	15.99	70
3	210100100027	2001-05-01	2001-05-31	16.99	70
4	210100100027	2002-05-01	2002-05-31	17.99	70
5	210100100030	1999-08-01	1999-08-31	28.99	70
6	210100100030	2000-08-01	2000-08-31	29.99	70

After Date transformation

Data Transformation Summary

After cleaning, some tables had a reduced number of observations and variables in some of the data sets. This is because the extracted data was dirty, and we had to delete some entries and remove unnecessary data.

For example, Table County had 3209 empty entries for **Province_Name** variable, 3363 and 3362 empty cells for **Region_Name** and **Region_type** columns respectively. The number of observations were reduced when data was cleaned.

```
> plot_missing_or_invalids_or_outliers(county, "Customer Details", 1)
    County_ID   County_Type   County_Name Province_Type Province_Name   Region_Name   Region_Type      State_ID       Country
            0           252           252          3208          3209          3363          3362             0          3304
```

The Staging Area now contains clean and validated data. After cleaning, we proceeded to complete the ETL process by loading the data to an Operational Data Store. This process can also be called Data Storing. This process is explained in detail in the next few slides.

Data Loading

Data is loaded into an operational data store for fast analysis of the data. This process makes access to the data very easy as well as the storage of data in various formats/groups to answer the required business questions.

Large volumes of data are also easily stored and accessible. There are various ways to store data but for the purpose of this project, we will be using the Dimensional Model because users can perform join queries to speed up data retrieval from various data sources.

The dimensional model also allows users to access the company's data from different points of view. This is also called 'multidimensional analysis'.

This data will be stored using the Star schema that is built from a Relational Database Management System as we have. The Star Schema uses Dimension tables that give descriptions with a centralised Fact table that contains metrics. The Dimension Tables created are shown in the figures below.

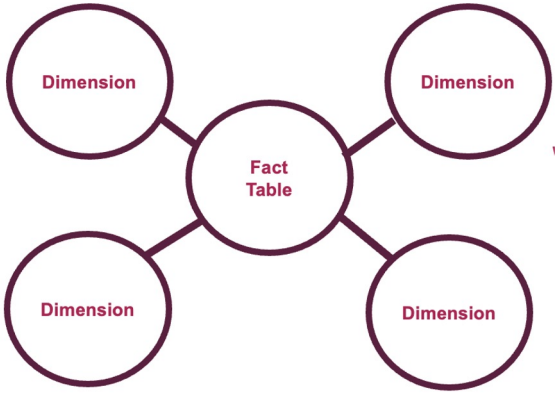


Figure 6

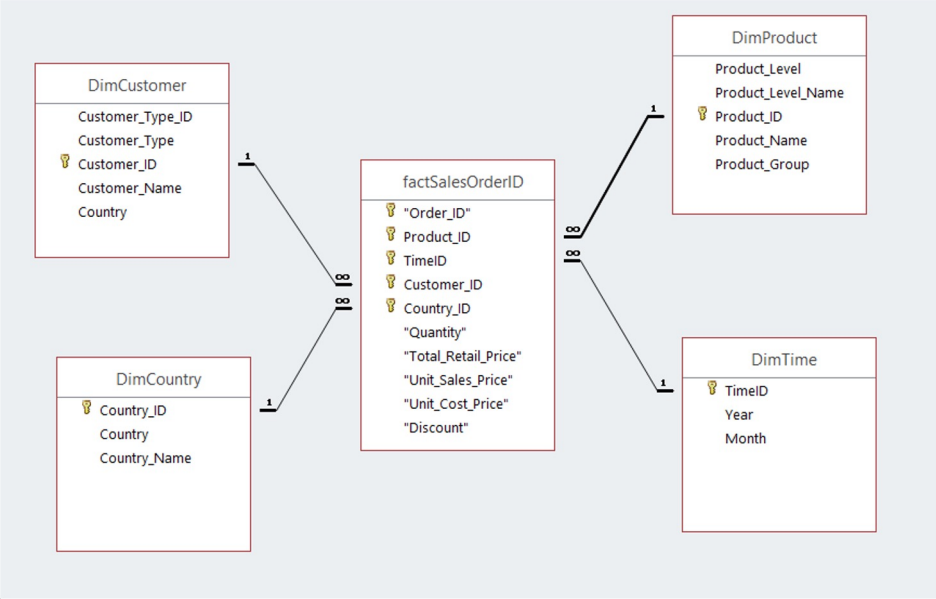
Data Loading

This data is stored in Hadoop file system where we are able to create the dimension and fact tables as seen in the figures above.

Hive is connected to R using the ODBC connector. In Hive, we are able run SQL queries to create dimension and a Sales fact table. The fact table is built specifically to answer our business questions outlined earlier. The dimensions tables are joined where necessary mostly using the left join so we can filter the outcome of the new tables.

SQL joins are used extensively as it helps to define the data the fact tables will contain. Once the Fact Tables are created, the business questions can be easily answered by moving around parameters.

For better understand and ease of interpretation, we have connected these data to tableau for Data visualisation .



Sales Data Mart

Data Loading

col_name	data_type	comment
country_id	bigint	
country	string	
country_name	string	

Country Dimension

col_name	data_type	comment
timeid	bigint	
year	int	
month	int	

Time Dimension

col_name	data_type	comment
customer_id	string	
customer_country	string	
customer_group	string	
customer_type	string	
customer_gender	string	
customer_age	string	
customer_name	string	
customer_firstname	string	
customer_lastname	string	
customer_birth_string	string	

Customer Dimension

col_name	data_type	comment
product_id	string	
product_line	string	
product_category	string	
product_group	string	
product_name	string	
supplier_country	string	
supplier_name	string	
supplier_id	string	

Product Dimension

Data Loading

col_name	data_type	comment
order_id	bigint	
order_date	string	
customer_id	bigint	
country	string	
product_id	bigint	
quantity	bigint	
total_retail_price	bigint	
unit_cost_price	bigint	
unit_sales_price	bigint	
discount	bigint	

Order Dimension

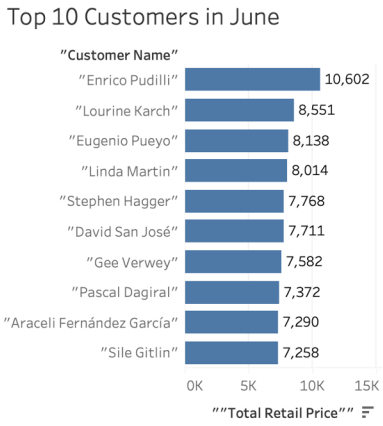
col_name	data_type	comment
order_id	int	
product_id	int	
timeid	int	
customer_id	int	
country_id	int	
quantity	bigint	
total_retail_price	bigint	
unit_sales_price	bigint	
unit_cost_price	bigint	
discount	bigint	

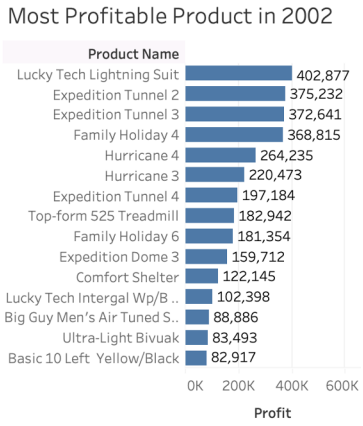
Fact Sales Table

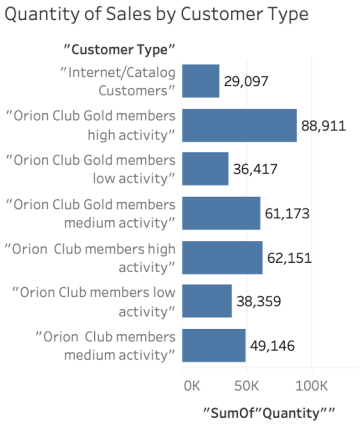
Data Visualisation

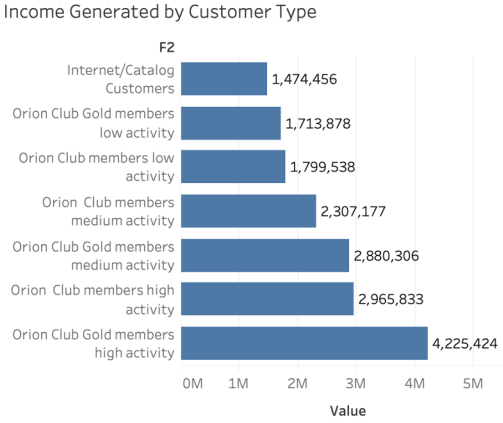
Data visualisation involves putting information into a visual context, such as a map or graph, to make it simpler for the human brain to grasp and draw conclusions from the data (Kristi M. et al, 2012). Making it simpler to spot patterns, trends, and outliers in huge data sets is the core objective of data visualisation.

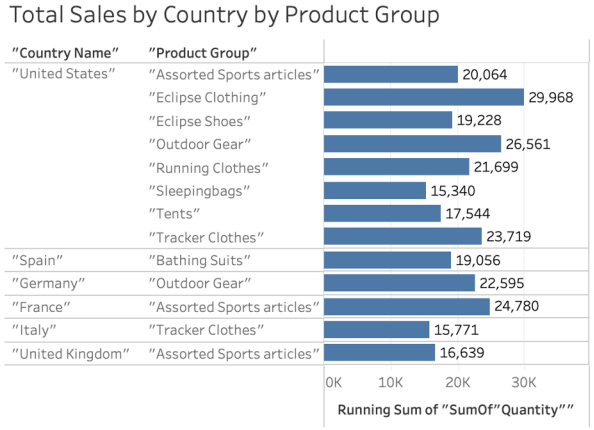
For visualisation, we have connected Hadoop to Tableau for a seamless transfer of data avoiding potential errors from downloading and uploading data. With Tableau, we are able to draw graphs as seen in the figures below for easier interpretation of the data. Analysis can more easily and accurately be drawn from the graphs.











Recommendations

From the results above, we can draw the following insights:

- Orion Club inactive members have contributed nothing to the sales and profit in year 2002.
- Orion Club Gold Members with High activity have accounted for the highest sales and profit in year 2002.
- The internet catalogue members have brought in the lowest sales as well as profit to the business.
- United States have accounted for the highest sales by quantity with eclipse clothing being the most sold product followed by France with the sales of Assorted sport articles.
- Orion's top 10 customers are also viewed and this data can be used to reward our customers.

Form these insights we will recommend:

- Focusing marketing to target online customers to boost sales.
- Also using coupons and discounts for online purchases can be a great way to increase sales.
- Inactive members can be given a gift/discount for reactivating their accounts. This can boost sales and increase the number of active buyers.
- Customers can be given loyalty cards that gain points for every purchase made and can be turned into cash. This will encourage sales as they will want to accumulate the reward points and subsequently purchase an item.

References

Golfarelli, Matte and Stefano Rizzi. (2009) “A Survey on Temporal Data Warehousing.” Int Journal of data Warehousing & Mining **5(1)**: 1–17.

Hackenberger BK. R software: unfriendly but probably the best. Croat Med J. 2020 Feb 29;61(1):66-68. doi: 10.3325/cmj.2020.61.66. PMID: 32118381; PMCID: PMC7063554.

Yin Huai et al. “Major Technical Advancements in Apache Hive”. In: Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data. SIGMOD ’14. New York, NY, USA: ACM, 2014, pp. 1235–1246. ISBN : 978-1-4503-2376-5. DOI:10 .1145/2588555 . 2595630.UR L:http ://doi . acm.org/ 10.1145/2588555.2595630.

Kristi Morton, Ross Bunker, Jock Mackinlay, Robert Morton, and Chris Stolte. 2012. Dynamic workload driven data integration in tableau. In Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data (SIGMOD '12). ACM, New York, NY, USA, 807-816. http://doi.acm.org/10.1145/2213836.2213961

Salim, Kelvin & Damayanti, Lorita & Puspita, Marisya & Liujaya, Steven & Girsang, Ganda. (2020). Data warehouse using Kimball approach in computer maniac. IOP Conference Series: Materials Science and Engineering. 725. 012099. 10.1088/1757-899X/725/1/012099

Appendix

For easy communication and sharing of data, A lot of communication was done via Microsoft Teams. This eased up communication and data sharing throughout the project. Meetings were scheduled and hed online to facilitate teamwork and group collaboration.
A shared drive was used for easy data sharing. A link to this shared drive has been attached below. It contains the necessary code and downloaded clean/transformed data.

https://drive.google.com/drive/folders/12kaeQMhcvL5xlK-jAheYeeZxEdxsYPfA?usp=sharing
Ab word document has also been attached with full code and additional information.